Parsing with an Extended Domain of Locality

John Carroll, Nicolas Nicolov, Olga Shaumyan, Martine Smets & David Weir
School of Cognitive and Computing Sciences
University of Sussex
Brighton, BN1 9QH, UK

Abstract

One of the claimed benefits of Tree Adjoining Grammars is that they have an extended domain of locality (EDOL). We consider how this can be exploited to limit the need for feature structure unification during parsing. We compare two wide-coverage lexicalized grammars of English, LEXSYS and XTAG, finding that the two grammars exploit EDOL in different ways.

1 Introduction

One of the most basic properties of Tree Adjoining Grammars (TAGs) is that they have an **extended domain of locality** (EDOL) (Joshi, 1994). This refers to the fact that the elementary trees that make up the grammar are larger than the corresponding units (the productions) that are used in phrase-structure rule-based frameworks. The claim is that in Lexicalized TAGs (LTAGs) the elementary trees provide a domain of locality large enough to state co-occurrence relationships between a lexical item (the **anchor** of the elementary tree) and the nodes it imposes constraints on. We will call this the **extended domain of locality hypothesis**.

For example, *wh*-movement can be expressed locally in a tree that will be anchored by a verb of which an argument is extracted. Consequently, features which are shared by the extraction site and the *wh*-word, such as case, do not need to be percolated, but are directly identified in the tree. Figure 1 shows a tree in which the case feature at the extraction site and the *wh*-word share the same value.[1]

[1]The anchor, substitution and foot nodes of trees are marked with the symbols $\diamond$, $\downarrow$ and $*$, respectively. Words in parenthesis are included in trees to provide examples of strings this tree can derive.

Much of the research on TAGs can be seen as illustrating how its EDOL can be exploited in various ways. However, to date, only indirect evidence has been given regarding the beneficial effects of the EDOL on parsing efficiency. The argument, due to Schabes (1990), is that benefits to parsing arise from lexicalization, and that lexicalization is only possible because of the EDOL. A parser dealing with a lexicalized grammar needs to consider only those elementary structures that can be associated with the lexical items appearing in the input. This can substantially reduce the effective grammar size at parse time. The argument that an EDOL is required for lexicalization is based on the observation that not every set of trees that can be generated by a CFG can be generated by a lexicalized CFG. But does the EDOL have any other more direct effects on parsing efficiency?

On the one hand, it is a consequence of the EDOL that wide-coverage LTAGs are larger than their rule-based counterparts. With larger elementary structures, generalizations are lost regarding the internal structure of the elementary trees. Since parse time depends on grammar size, this could have an adverse effect on parsing efficiency. However, the problem of grammar size in TAG has to some extent been addressed both with respect to grammar encoding (Evans et al., 1995; Candito, 1996) and parsing (Joshi and Srinivas, 1994; Evans and Weir, 1998).

On the other hand, if the EDOL hypothesis holds for those dependencies that are being checked by the parser, then the burden of passing feature values around during parsing will be less than in a rule-based framework. If *all* dependencies that the parser is checking can be stated directly within the elementary structures of the grammar, they do not need to be computed dynamically during the parsing process by means of feature percolation. For example, there is no need to use a slash feature to establish filler-gap dependencies over unbounded distances across the tree if the EDOL

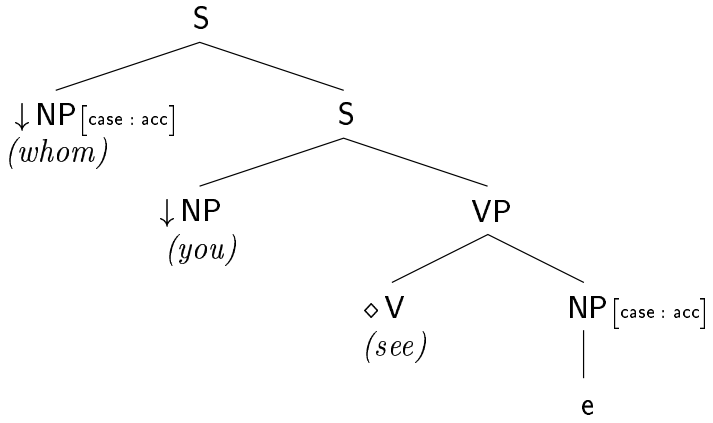

Figure 1: Localizing a filler-gap dependency

makes it possible for the gap and its filler to be located within the same elementary structure.

This paper presents an investigation into the *extent* to which the EDOL reduces the need for feature passing in two existing wide-coverage grammars: the XTAG grammar (XTAG-Group, 1995), and the LEXSYS grammar (Carroll et al., 1998). It can be seen as an evaluation of how well these two grammars make use of the EDOL hypothesis with respect to those dependencies that are being checked by the parser.

2 Parsing Unification-Based Grammars

In phrase-structure rule-based parsing, each rule corresponds to a *local tree*. A rule is applied to a sequence of existing contiguous constituents, if they are compatible with the daughters. In the case of context-free grammar (CFG), the compatibility check is just equality of atomic symbols, and an instantiated daughter is merely the corresponding sub-constituent.

However, unification-based extensions of phrase-structures grammars are used because they are able to encode local and non-local syntactic dependencies (for example, subject-verb agreement in English) with re-entrant features and feature percolation, respectively. Constituents are represented by DAGs (directed acyclic graphs), the compatibility check is unification, and it is the result of each unification that is used to instantiate the daughters. Graph unification based on the UNION-FIND algorithm has time complexity that is near-linear in the number of feature structure nodes in the inputs (Huet, 1975; Ait-Kaci, 1984); however, feature structures in wide-coverage grammars can contain hundreds of nodes (see e.g., HPSG (Pollard and Sag, 1994)), and since unification is a primitive operation the overall number of unification attempts during parsing can be very large. Unification therefore has a substantial practical cost.

Efficient graph unification must also ensure that it does not destructively modify the input structures, since the same rule may be used several times within a single derivation, and also the same constituent may be used within different partial analyses with features instantiated in different ways. Copying input feature structures in their entirety before each unification would solve this problem, but the space usage renders this approach impractical. Therefore, various 'quasi-destructive' algorithms (Karttunen, 1986; Kogure, 1990; Tomabechi, 1991) and algorithms using 'skeletal' DAGs with updates (Pereira, 1985; Emele, 1991) have been proposed, which attempt to minimize copying. But even with good implementations of the best of these improved algorithms, parsers designed for wide-coverage unification-based phrase-structure grammars using large HPSG-style feature graphs spend around 85–90% of their time unifying and copying feature structures (Tomabechi, 1991), and may allocate in the region of 1–2 Mbytes memory while parsing sentences of only eight words or so (Flickinger, p.c.). Although comfortably within main memory capacities of current workstations, such large amounts of short-term storage allocation overflow CPU caches, and storage management overheads become significant.

In the case of unification-based LTAG the situation is even more problematic. Elementary structures are larger than productions, and the potential is that the parser will have to make copies of entire trees and associated feature structures. Furthermore, the number of trees that an LTAG

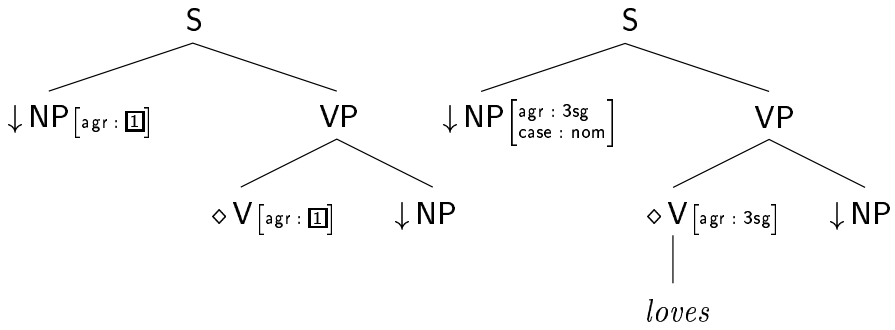

Figure 2: Unanchored and anchored trees localizing subject/verb agreement

parser must consider tends to be far larger than the number of rules in a corresponding phrase-structure grammar. On the other hand, the EDOL has the potential to eliminate some or all of feature percolation, and in the remainder of this section, we explain how.

An LTAG consists of a set of unanchored trees such as the one shown on the left of Figure 2. This shows a tree for transitive verbs where subject/verb agreement is captured directly with re-entrancy between the value of agr feature structures at the anchor (verb) node and the subject node. Notice the re-entrancy between the anchor node and the substitution node for the subject. However these are not the trees that the parser works with; the parser is given trees that have been *anchored* by the morphologically analysed words in the input sentence. For example, the tree shown on the right is the result of anchoring the tree shown on the left with the word *loves*. Anchoring instantiates the agr feature of the anchor node as 3sg which has the effect (due to the re-entrancy in the unanchored tree) of instantiating the agr feature at the subject node as 3sg.

Anchored elementary trees are translated by the parser into a sequence of what we will refer to as **parser actions**. For example, once the tree shown on the right of Figure 2 has been associated with the word *loves* in the input, it can be recognized with a sequence of parser actions that involve finding a NP constituent on the right (corresponding to the object), possibly performing adjunctions at the VP node, and then finding another NP constituent on the left (corresponding to the subject). We say that the two NP substitution nodes and the VP node are the sites of parser actions in this tree. Problems arise, and the EDOL hypothesis is violated, when there is a dependence between different parser actions.

The EDOL hypothesis states that elementary trees provide a domain of locality large enough to state co-occurrence relationship between the anchor of the tree and the nodes it imposes constraints on. If all dependencies relevant to the parser can be captured in this way then, once an elementary tree has been anchored by a particular lexical item, the settings of feature values at all of the dependent nodes will have been fixed, and no feature percolation can occur. Each unification is a purely local operation with no repercussions on the rest of the parsing process. No copying of feature structures is required, so memory usage is greatly reduced, and complex quasi-destructive algorithms with their associated computational overheads can be dispensed with.

Note that, although feature *percolation* is eliminated when the EDOL hypothesis holds, the feature structure at a node can still change. For example, substituting a tree for a proper noun at the subject position of the tree in Figure 2 would cause the feature structure at the node for the subject to include pn:+. This, however, does not violate the EDOL hypothesis since this feature is not coreferenced with any other feature in the tree.

3 Analysis of two wide-coverage grammars

As we have seen, the EDOL of LTAGs makes it possible, at least in principle, to locally express dependencies which cannot be localized in a CFG-based formalism. In this section we consider two existing grammars: the XTAG grammar, a wide-coverage LTAG, and the LEXSYS grammar, a wide-coverage D-Tree Substitution Grammar (Rambow et al., 1995). For each grammar we investigate the extend to which they do not take full advantage of the EDOL and require percolation of features at parse time.

There are a number of instances in which dependencies are not localized in the XTAG grammar, most of which involve auxiliary trees. There are

three types of auxiliary trees: predicative, modifier and coordination auxiliary trees. In predicative auxiliary trees the anchor is also the head of the tree and becomes the head of the tree resulting from the adjunction. In modifier auxiliary trees, the anchor is not the head of the tree, and the subtree headed by the anchor usually plays a role of adjunct in the resulting tree. Coordination auxiliary trees are similar to modifier auxiliary trees in that they are anchored by the conjunction which is not the head of the phrase. One of the conjoined nodes is a foot node, the other one a substitution node.

3.1 Modifier Auxiliary Trees

In modifier auxiliary trees — an example of which is shown in Figure 3[2] — the feature values at the root and foot nodes are set by the node at which the auxiliary tree is adjoined, and have to be percolated between the foot node and the root node. The LEXSYS grammar adopts a similar account of modification.

From a parsing point of view, this does not result in the need for feature percolation: only the foot node of the modifier tree is the site of a parser action, and the root node is ignored by the process that interprets the tree for the parser.

3.2 Coordination Auxiliary Trees

An example of an XTAG coordination auxiliary tree is shown on the left of Figure 4. This case is different from the modification case since features of the substitution node have to be identical to features of the foot node (which will match those at the adjunction site). From a parsing point of view these nodes are both the sites of actions, resulting in the need for feature percolation. For example, for the NP coordination tree shown in Figure 4, if one of the conjuncts is a *wh*-phrase, the other conjunct must be a *wh*-phrase too, as in *who or what did this?* but **John and who did this?* The *wh*-feature has to be percolated between the two nodes on each side of the conjunction.

In the LEXSYS grammar, a coordination tree is anchored by a head of the tree, not by the conjunction. To illustrate (see the tree on the right of Figure 4), NP-coordination trees are anchored by a noun, and features such as wh and case are ground during anchoring. As a result, there is no need for passing of these features in the coordination trees of the LEXSYS grammar.

[2]All examples relating to the XTAG grammar come from the XTAG report (XTAG-Group, 1995). They have been simplified to the extent that only details relevant to the discussion are included.

As for agreement features, there are two cases to consider: if the conjunction is *and*, the number feature of the whole phrase is *plural*; if the conjunction is *or*, the number feature is the same as the last conjunct's (XTAG-Group, 1999). In both the XTAG and LEXSYS grammars, this is achieved by having separate trees for each type of conjunction.

3.3 Predicative Auxiliary Trees

In the XTAG grammar, subject raising and auxiliary verbs anchor auxiliary trees rooted in VP, without a subject[3]; they can be adjoined at the VP node of any compatible verb tree. With this arrangement, subject-verb agreement must be established dynamically. The agr feature of the NP subject must match the agr feature of whichever VP ends up being highest at the end of the derivation. In Figure 5, the *bought* tree has been anchored in such a way that adjunction at the VP node is obligatory, since a matrix clause cannot have mode:ppart.[4] When the tree for *has* is adjoined at the VP node the agr features of the subject will agree with those of *bought*. The feature structure at the root of the tree for *has* is unified with the upper feature structure at the VP node of the tree for *bought*, and the feature structure at the foot of the tree for *has* is unified with the lower feature structure at the VP node of the tree for *bought*. The foot node of the *has* tree is the VP node on the frontier of the tree. Note that even after the tree has been anchored, re-entrancy of features occurs in the tree.

Thus, there are two sites in the tree for *bought* (the subject NP node and the VP node) at which parser actions will take place (substitution and adjunction, respectively) such that a dependency between the values of the features at these two nodes must be established by the parser.

The situation is similar for case assignment (also shown in the Figure 5): the value of a feature ass-case (the assign case feature) on the highest VP is coreferred with the value of the feature case on the subject NP. For finite verbs, the value of the feature ass-case is determined by the mode of the verb. For infinitive verbs, case is assigned in various ways, the details of which are not relevant to the discussion here. The subject is in the nominative case if the verb is finite, and in the accusative otherwise. As with the agr feature, the value of the case feature cannot be instantiated

[3]To allow for long distance dependency, subject raising verbs must anchor an auxiliary tree, with identical root and foot nodes, a VP.

[4]Unifying the two feature structures at the VP node would cause a matrix clause to have mode:ppart.

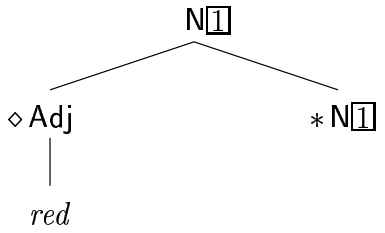

Figure 3: XTAG example of modifying auxiliary tree

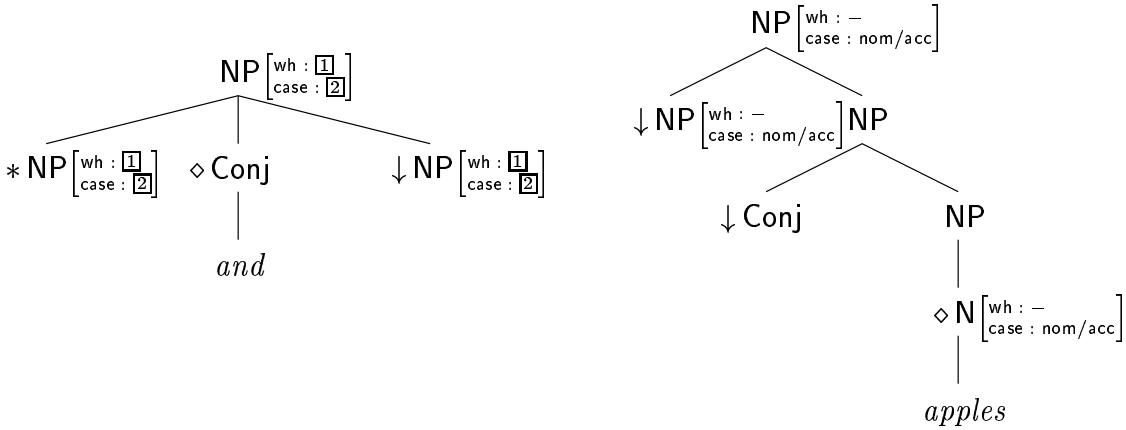

Figure 4: Coordination in XTAG (left) and LEXSYS (right)

in the anchored elementary tree of the main verb because auxiliary verb trees can be adjoined.

The same observations apply to the XTAG treatment of copula with predicative categories such as an adjective. As shown in Figure 6, these predicative AP trees have a subject but no verb; trees for raising verbs or the copula can be adjoined into them. As in the previous example, the agr features of the verb and subject cannot be instantiated in the elementary tree because the verb and its subject are not present in the same tree.

From the examples we have seen, it appears that the XTAG grammar does not take full advantage of the EDOL with respect to a number of syntactic features, for example those relating to agreement and case. The LEXSYS grammar takes a rather different approach to phenomena that XTAG handles with predicative auxiliary trees.

The LEXSYS grammar has been designed to localize *syntactic* dependencies in elementary trees. As in the XTAG grammar, unbounded dependencies between gap and filler are localized in elementary trees; but unlike the XTAG grammar, other types of syntactic dependencies, such as agreement, are also localized. All finite verbs, including auxiliary and raising verbs, anchor a tree rooted in S, and thus are in the same tree as the subject with which they agree. An example involving finite verbs is shown in Figure 7. Since verb trees cannot be substituted between the subject and the verb, the agr feature can be grounded when elementary trees are anchored, rather than during the derivation. The case feature of the subject can be specified even in the *unanchored* elementary tree: in trees for finite verbs the subject has nominative case; in trees for *for ... to* clauses it has accusative case.

As can be seen from the tree on the right of Figure 7, subject raising and auxiliary verbs are rooted in S and take a VP complement. So the sentence *He seems to like apples* is produced by substituting a VP-rooted tree for *to like* into a tree for *seems*.

Thus, for all three trees shown in Figure 7, once anchoring has taken place, all of the syntactic features being checked by the parser are grounded. Hence, the parser does not have to check for dependencies between the parser actions taking place at different sites in the tree.

3.4 Semantic Dependencies

There are many examples where the XTAG grammar, but not the LEXSYS grammar, localizes semantic dependencies: for example, dependencies

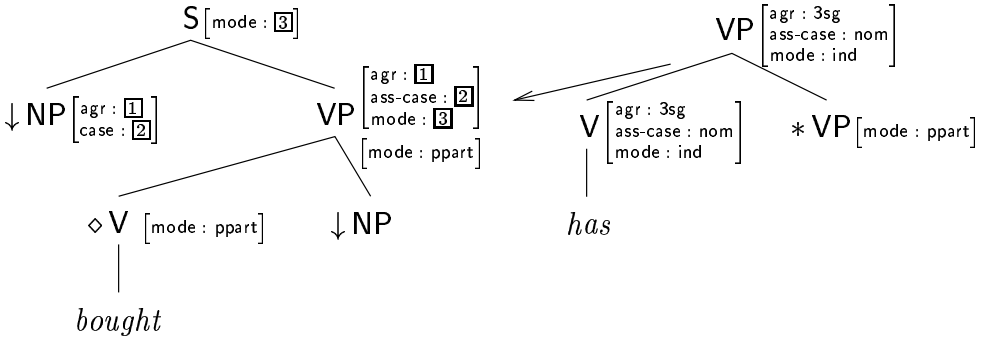

Figure 5: XTAG example with a raising verb

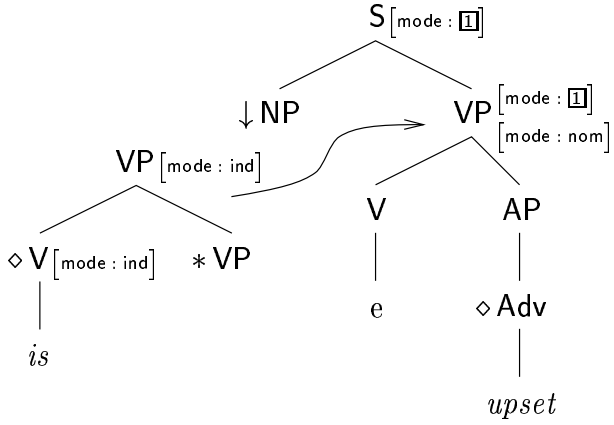

Figure 6: XTAG example of a predicative adjective

between an adjective and its subject. As shown in Figure 6, in XTAG the predicative adjective and its subject are localized in the same elementary tree, and selectional restrictions *can* be locally imposed by the adjective on the subject without the need for feature percolation. On the other hand, in the LEXSYS grammar, the dependency between *upset* and *he* in *he looks upset* could not be checked during parsing without the use of feature passing between the subject and AP node of the tree in the middle of Figure 7.

3.5 Percolation of Features in LEXSYS

This section considers a limited number of cases where it appears that it is not possible to set all syntactic features by anchoring an elementary tree.

When two nodes other than the anchor of the tree are syntactically dependent, feature values may have to be percolated between these nodes (the anchor does not determine the value of these features). For example, in English adjectives that can have S subjects determine the verb form of the subject. Hence, in Figure 8, the verb form feature of the subject is not determined by the anchor of the tree (the verb) but by the complement of the anchor (the adjective). The verb form feature must therefore be percolated from the adjective phrase to the subject.

The XTAG grammar localizes this dependency (see Figure 6). However, as we have seen, agreement features are *not* localized in this analysis. The problem then is that it does not seem to be possible to localize all syntactic features in this case.

Feature percolation is also required in the LEXSYS grammar for prepositional phrases which contain a *wh*-word, because the value of the wh feature is not set by the anchor of the phrase (the preposition) but by the complement (as in *these reports, the wording on the covers of which has caused so much controversy, are to be destroyed*[5]). The value of the feature wh is set by the NP-complement, and percolated to the root of the PP.

4 Conclusions

In XTAG both syntactic and semantic features are considered during parsing, whereas in the LEXSYS

[5]Example borrowed from Gazdar et al. 1985.

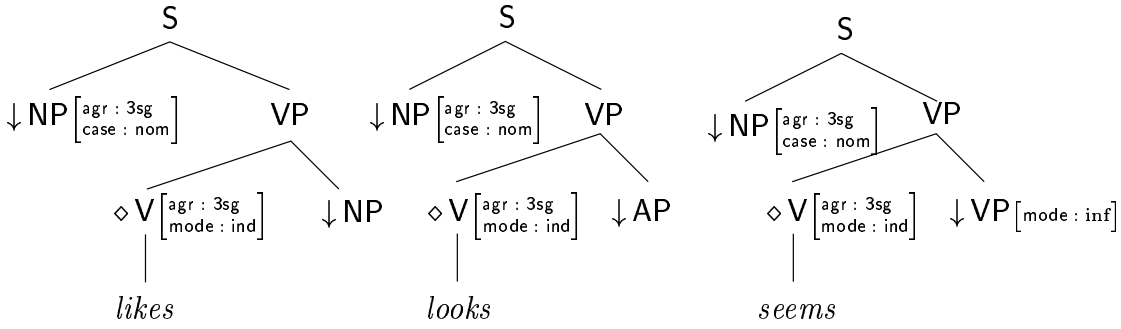

Figure 7: LEXSYS example for case and agr features

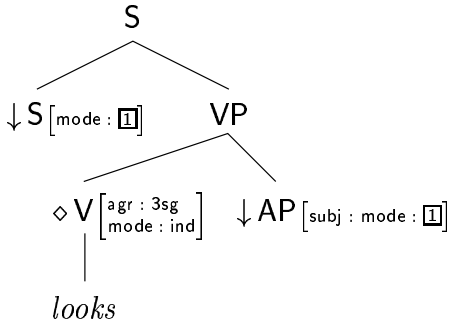

Figure 8: LEXSYS example for subject/adjective syntactic dependency

system only syntactic dependencies are considered during parsing; semantic dependencies are left for a later processing phase. The LEXSYS parser returns a complete set of all syntactically well-formed derivations. Semantic information can be recovered from derivation trees and then processed as desired.

From a processing point of view, the XTAG and LEXSYS grammars are examples that show that the checking of dependencies involves a trade-off.[6] On the one hand, a greater number of parses may be returned if the only dependencies checked are syntactic, since possible violations of semantic dependencies are ignored. On the other hand, as we have seen in this paper, there are potentially substantial benefits to parsing efficiency if all dependencies that the parser is checking can be localized with the EDOL. It is too early to say how best to make the trade-off, but by comparing the way that the XTAG and LEXSYS grammars exploit the EDOL, we hope to have shed some light on the role that the EDOL can play with respect to parsing efficiency.

[6]These are both grammars for English. Hence, whether the conclusions we draw apply to other languages is outside the scope of the present work.

5 Acknowledgements

This work is supported by UK EPSRC project GR/K97400 and by an EPSRC Advanced Fellowship to the first author. We would like to thank Roger Evans, Gerald Gazdar & K. Vijay-Shanker for helpful discussions.

References

Hassan Ait-Kaci. 1984. *A Lattice Theoretic Approach to Computation Based on a Calculus of Partially Ordered Type Structures*. Ph.D. thesis, Department of Computer and Information Science, University of Pennsylvania, Philadelphia, PA.

Marie-Hélène Candito. 1996. A principle-based hierarchical representation of LTAGs. In *Proceedings of the 16th International Conference on Computational Linguistics*, Copenhagen, Denmark, August.

John Carroll, Nicolas Nicolov, Olga Shaumyan, Martine Smets, and David Weir. 1998. The LEXSYS Project. In *Proceedings of the Fourth International Workshop on Tree Adjoining Grammars and Related Frameworks*, pages 29–33.

Martin Emele. 1991. Unification with lazy non-redundant copying. In *Proceedings of the 29th Meeting of the Association for Computational Linguistics*, pages 323–330, Berkeley, CA.

Roger Evans and David Weir. 1998. A structure-sharing parser for lexicalized grammars. In *Proceedings of the 36th Meeting of the Association for Computational Linguistics and the 17th International Conference on Computational Linguistics*, pages 372–378.

Roger Evans, Gerald Gazdar, and David Weir. 1995. Encoding lexicalized Tree Adjoining Grammars with a nonmonotonic inheritance hierarchy. In *Proceedings of the 33rd Meeting of the Association for Computational Linguistics*, pages 77–84.

G. P. Huet. 1975. A unification algorithm for typed λ-calculus. *Theoretical Computer Science*, 1:27–57.

Aravind Joshi and B. Srinivas. 1994. Disambiguation of super parts of speech (or supertags): Almost parsing. In *Proceedings of the 15th International Conference on Computational Linguistics*, pages 154–160.

Aravind Joshi. 1994. Preface to special issue on Tree-Adjoining Grammars. *Computational Intelligence*, 10(4):vii–xv.

Lauri Karttunen. 1986. D-PATR: A development environment for unification-based grammars. In *Proceedings of the 11th International Conference on Computational Linguistics*, pages 74–80, Bonn, Germany.

Kiyoshi Kogure. 1990. Strategic lazy incremental copy graph unification. In *Proceedings of the 13th International Conference on Computational Linguistics*, pages 223–228, Helsinki.

Fernando Pereira. 1985. A structure-sharing representation for unification-based grammar formalisms. In *Proceedings of the 23rd Meeting of the Association for Computational Linguistics*, pages 137–144.

Carl Pollard and Ivan Sag. 1994. *Head-Driven Phrase Structure Grammar.* University of Chicago Press, Chicago.

Owen Rambow, K. Vijay-Shanker, and David Weir. 1995. D-Tree Grammars. In *Proceedings of the 33rd Meeting of the Association for Computational Linguistics*, pages 151–158.

Yves Schabes. 1990. *Mathematical and Computational Aspects of Lexicalized Grammars.* Ph.D. thesis, Department of Computer and Information Science, University of Pennsylvania.

Hideto Tomabechi. 1991. Quasi-destructive graph unification. In *Proceedings of the 29th Meeting of the Association for Computational Linguistics*, pages 315–322, Berkeley, CA.

The XTAG-Group. 1995. A lexicalized Tree Adjoining Grammar for English. Technical Report IRCS Report 95-03, The Institute for Research in Cognitive Science, University of Pennsylvania.

The XTAG-Group. 1999. A lexicalized Tree Adjoining Grammar for English. Technical Report `http://www.cis.upenn.edu/~xtag/tech-report/tech-report.html`, The Institute for Research in Cognitive Science, University of Pennsylvania.